HW 1: Introduction to VerySimpleCPU

Objective

Introduce the concepts and general practices for programming VerySimpleCPU using our Instruction Set Simulator (ISS).

Background

B1: VerySimpleCPU ISS

VerySimpleCPU.exe is an ISS, that is, simulation model.

- Takes a program file, coded in VerySimpleCPU assembly language, as input,
- Reads VerySimpleCPU instructions,
- Executes the instructions, and
- Writes results on to the command window as well as text files.

Provides a debugging environment for VerySimpleCPU programmers.

B2: VerySimpleCPU Instruction Set

Please look in "**InstructionSet**" folder. Make sure you **go through the README file**, which contains the detailed explanations of the instructions. Do not forget to look at the pictures in the same folder.

B3: Operating System Requirement

For those of you running a Windows Operating System (OS), use "VerySimpleCPU.exe" in "VSCPU-Sim" folder. Those who have Linux or Mac, there is also http://verysimplecpu.org/ which can be useful, but your files **must be runnable without errors with the simulators in the folders**.

B4: Syntax

In Hi C, your syntax should have the appropriate **indentation (tab) as necessary**. In Low C, there should be **no indentation**, every single line of code should **correspond to a VSCPU instruction**. For example:

Low C	Assembly
tmp = a;	0: CP 100 101 //consider address 100 as tmp, 101 as a
tmp = tmp < a;	1: LT 100 101

Please remember that we have only two operands, namely A and B in VSCPU instructions. You **cannot** have "isSmaller = tmp < a;" in Low C.

Hi C	Low C	Assembly

isSmaller = tmp < a;	isSmaller = tmp;
	isSmaller = isSmaller < a;
		0: CP 110 100 //consider 110 as isSmaller
		1: LT 110 101

In VSCPU Assembly code, the syntax must be in the form of one of the two below.

Address: OPCODE A B
Address: data

What To Do

Make sure you **watch the video before doing the homework**: https://youtu.be/4CSSaXYq7dU

Part 1

In Part 1 of this homework, you will run your first VerySimpleCPU assembly program that finds the maximum of the two numbers in addresses **100, 101** and writes the result into address 110. In Part 2, you modify the program so that it does something else and then you run it again.

1. Create a workspace for EE321 Homeworks
    a. Create a new folder and name it "EE321workspace"
    b. Download "HW1" folder from LMS
    c. Download "VSCPU-Sim" or "VSCPU-Sim-Linux-Mac" folder depending on your OS
2. Open and analyze the two C files with a text editor. (I suggest you download notepad++.)
3. Open "hw1_part1_max.asm" then analyze and compare it with "hw1_part1_max_low.c"
4. Open the command prompt
5. Type "cd" with file path which is "..\EE321workspace\VSCPU-Sim" (or VSCPU-Sim-Linux-Mac) (i.e. cd C:\Users\student\Desktop\EE321workspace\VSCPU-Sim)
6. Type "VerySimpleCPU.exe ../HW1/hw1_part1_max.asm r > log" then press ENTER
7. Type "exit" then press ENTER again
8. Open the log file and look at it. The ISS displays memory location before and after every instruction.
9. Look at "memoutd.txt". It displays the final memory contents. Look at location 110 and make sure it is the biggest among *100 and *101.

Part 2

1. Rewrite the two C codes and the .asm in part1, hw1_part1_max_hi.c, hw1_part1_max_low.c, and hw1_part1_max.asm so that the program finds the minimum of 3 numbers at *730, *731, *732 and writes to *700. Name those new files hw1_part2_min3_hi.c, hw1_part2_min3_low.c and hw1_part2_min3.asm.
2. Run your program and check the result through memoutd.txt.
3. If it does not work correctly, debug through the "log" file.

Submission

- Submit the following files in LMS under the assignment HW1. Do not zip your files, upload them directly on LMS!
    - hw1_part2_min3_hi.c
    - hw1_part2_min3_low.c
    - hw1_part2_min3.asm